////// CRM Branch Master Service Implementation

package com.maan.crm.service.impl;

import java.awt.image.BufferedImage;

import java.io.File;

import java.io.IOException;

import java.math.BigDecimal;

import java.nio.file.Files;

import java.nio.file.Path;

import java.nio.file.Paths;

import java.text.DecimalFormat;

import java.text.SimpleDateFormat;

import java.util.ArrayList;

import java.util.Calendar;

import java.util.Date;

import java.util.List;

import java.util.Random;

import javax.imageio.ImageIO;

import org.apache.commons.io.FileUtils;

import org.apache.commons.io.FilenameUtils;

import org.apache.commons.lang3.StringUtils;

import org.apache.logging.log4j.LogManager;

import org.apache.logging.log4j.Logger;

import org.modelmapper.ModelMapper;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.beans.factory.annotation.Value;

import org.springframework.data.domain.Pageable;

import org.springframework.data.domain.Page;

import org.springframework.data.domain.PageRequest;

import org.springframework.data.domain.Sort;

import org.springframework.security.web.server.context.ReactorContextWebFilter;

import org.springframework.stereotype.Service;

import org.springframework.transaction.annotation.Transactional;

import org.springframework.web.multipart.MultipartFile;

import net.coobird.thumbnailator.Thumbnails;

import com.google.gson.Gson;

import com.maan.crm.bean.CrmBranchMaster;

import com.maan.crm.error.Error;

import com.maan.crm.repository.CrmBranchMasterRepository;

import com.maan.crm.repository.LoginMasterRepository;

import com.maan.crm.req.BranchFiles;

import com.maan.crm.req.CrmBranchMasterPaginationReq;

import com.maan.crm.req.CrmBranchMasterSaveReq;

import com.maan.crm.res.CrmBranchMasterRes;

import com.maan.crm.res.CrmBranchMasterSuccessRes;

import com.maan.crm.service.CrmBranchMasterService;

import net.coobird.thumbnailator.Thumbnails;

import java.util.concurrent.Callable;

/\*\*

\* <h2>CrmBranchMasterServiceimpl</h2>

\*/

@Service

@Transactional

public class CrmBranchMasterServiceImpl implements CrmBranchMasterService {

@Autowired

private CrmBranchMasterRepository repository;

@Autowired

private LoginMasterRepository loginRepo;

Gson json =new Gson();

private Logger log = LogManager.getLogger(CrmBranchMasterServiceImpl.class);

SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy");

String pattern = "#####0";

DecimalFormat decimalFormat = new DecimalFormat(pattern);

@Value("${file.upload-dir2}")

private String directoryPath;

@Value("${common.file.path}")

private String compressedPath;

//// Validation

@Override

public List<Error> validateBranchMaster(CrmBranchMasterSaveReq req) {

// TODO Auto-generated method stub

// TODO Auto-generated method stub

List<Error> errors = new ArrayList<Error>();

try {

if (StringUtils.isNotBlank(req.getBranchCode())) {

if (!StringUtils.isNumeric(req.getBranchCode())) {

errors.add(new Error("01", "Branch Code", "Please Enter Valid Branch Code "));

}

if (req.getBranchName()== null || StringUtils.isBlank(req.getBranchName())) {

errors.add(new Error("02", "Branch Name", "Please Enter Branch Name"));

} else if (req.getBranchName().length() > 50) {

errors.add(new Error("02", "Branch Name", "Please Enter Branch Name within 50 Characters"));

}

if (req.getRegionCode()== null || StringUtils.isBlank(req.getRegionCode())) {

errors.add(new Error("03", "Region Code", "Please Enter Region Code"));

} else if (req.getRegionCode().length() > 10) {

errors.add(new Error("03", "Region Code", "Please Enter Region Code within 10 Characters"));

}

if (req.getInsCompanyId()== null || StringUtils.isBlank(req.getInsCompanyId())) {

errors.add(new Error("04", "Insurance Company Id", "Please Enter Insurance Company Id"));

}

if (req.getAddress1()== null || StringUtils.isBlank(req.getAddress1())) {

errors.add(new Error("05", "Address 1", "Please Enter Address 1"));

} else if (req.getAddress1().length() > 75) {

errors.add(new Error("05", "Address 1", "Please Enter Address 1 within 75 Characters"));

}

if (req.getAddress2()== null || StringUtils.isBlank(req.getAddress2())) {

errors.add(new Error("06", "Address 2", "Please Enter Address 2"));

} else if (req.getAddress2().length() > 75) {

errors.add(new Error("06", "Address 2", "Please Enter Address 2 within 75 Characters"));

}

if (req.getAddress3()== null || StringUtils.isBlank(req.getAddress3())) {

errors.add(new Error("07", "Address 3", "Please Enter Address 3"));

} else if (req.getAddress3().length() > 75) {

errors.add(new Error("07", "Address 3", "Please Enter Address 3 within 75 Characters"));

}

if (req.getCity()== null || StringUtils.isBlank(req.getCity())) {

errors.add(new Error("08", "City", "Please Enter City"));

} else if (req.getCity().length() > 25) {

errors.add(new Error("08", "City", "Please Enter City within 25 Characters"));

}

if (req.getCountry()== null || StringUtils.isBlank(req.getCountry())) {

errors.add(new Error("09", "Country", "Please Enter Country"));

} else if (req.getCountry().length() > 25) {

errors.add(new Error("09", "Country", "Please Enter Country within 25 Characters"));

}

if (req.getPhone()== null || StringUtils.isBlank(req.getPhone())) {

errors.add(new Error("10", "Phone", "Please Enter Phone Number"));

} else if (req.getPhone().length() > 15) {

errors.add(new Error("10", "Phone", "Please Enter Phone Number within 15 Characters"));

}

if (req.getEmail()== null || StringUtils.isBlank(req.getEmail())) {

errors.add(new Error("11", "Email", "Please Enter Email"));

} else if (req.getEmail().length() > 40) {

errors.add(new Error("11", "Email", "Please Enter Email within 40 Characters"));

}

if (req.getBelongingType()== null || StringUtils.isBlank(req.getBelongingType())) {

errors.add(new Error("12", "Belonging Type", "Please Enter Belonging Type"));

} else if (req.getBelongingType().length() > 1) {

errors.add(new Error("12", "Belonging Type", "Please Enter Belonging Type within 1 Character"));

}

if (req.getBelongingBranch() ==null || StringUtils.isBlank(req.getBelongingBranch())) {

errors.add(new Error("13", "Belonging Branch", "Please Enter Belonging Branch"));

} else if (req.getBelongingBranch().length() > 1) {

errors.add(new Error("12", "Belonging Branch", "Please Enter Belonging Branch within 10 Characters"));

}

if (req.getCoreAppCode().length() > 100) {

errors.add(new Error("13", "Core App Code", "Please Enter Core App Code within 100 Characters"));

}

if (StringUtils.isBlank(req.getBranchCode()) && StringUtils.isNotBlank(req.getCoreAppCode())) {

Long branchCount = repository.countByCoreAppCode(req.getCoreAppCode());

if (branchCount> 0)

errors.add(new Error("13", "Core App Code",

"Please Enter New Core App Code " + req.getCoreAppCode() + " is exist already"));

} else if (StringUtils.isNotBlank(req.getBranchCode()) && StringUtils.isNotBlank(req.getCoreAppCode())) {

Long branchandCoreCount = repository.countByBranchCodeNotAndCoreAppCode(Integer.valueOf(req.getBranchCode()),

req.getCoreAppCode());

if (branchandCoreCount> 0)

errors.add(new Error("13", "Core App Code",

"Please Enter New Core App Code " + req.getCoreAppCode() + " is exist already"));

}

if (req.getStatus() == null || StringUtils.isBlank(req.getStatus())) {

errors.add(new Error("14", "Status", "Please Enter Status"));

} else if (req.getStatus().length() > 1) {

errors.add(new Error("14", "Status", "Please Enter Status within 1 Character"));

} else if (!(req.getStatus().equals("Y") || req.getStatus().equals("N"))) {

errors.add(new Error("14", "Status", "Please Enter Status Properly Yes-Y or No-N"));

}

Calendar cal = Calendar.getInstance();

cal.add(Calendar.DATE, -1);

Date yesterday = cal.getTime();

Date a1 = sdf.parse(req.getEffectiveDate());

if (req.getEffectiveDate() == null || StringUtils.isBlank(req.getEffectiveDate().toString())) {

errors.add(new Error("15", "Effective Date", "Please Enter Effective Date"));

} else if (a1.before(yesterday)) {

errors.add(new Error("15", "EffectiveDate", "Please Enter Future Date as EffectiveDate"));

} else if (!req.getEffectiveDate().matches("([0-9]{2})/([0-9]{2})/([0-9]{4})")) {

errors.add(new Error("15", "EffectiveDate",

"Effective Date format should be dd/MM/yyyy only allowed . Example :- 15/12/2022"));

}

if (req.getRemarks().length() > 100) {

errors.add(new Error("16", "Remarks", "Please Enter Remarks within 100 Characters"));

}

if (req.getFax() == null || StringUtils.isBlank(req.getFax())) {

errors.add(new Error("17", "Fax", "Please Enter Fax"));

} else if (req.getFax().length() > 25) {

errors.add(new Error("17", "Fax", "Please Enter Fax within 25 Characters"));

}

if (req.getHeaderImg() == null || StringUtils.isBlank(req.getHeaderImg())) {

errors.add(new Error("18", "Header Image", "Please Enter Header Image"));

} else if (req.getHeaderImg().length() > 50) {

errors.add(new Error("18", "Header Image", "Please Enter Header Image within 50 Characters"));

}

if (req.getFooterImg() == null || StringUtils.isBlank(req.getFooterImg())) {

errors.add(new Error("19", "Footer Image", "Please Enter Footer Image"));

} else if (req.getFooterImg().length() > 50) {

errors.add(new Error("19", "Footer Image", "Please Enter Footer Image within 50 Characters"));

}

if (req.getSignImg()== null || StringUtils.isBlank(req.getSignImg())) {

errors.add(new Error("20", "Sign Image", "Please Enter Sign Image"));

} else if (req.getSignImg().length() > 50) {

errors.add(new Error("20", "Sign Image", "Please Enter Sign Image within 50 Characters"));

}

if (req.getStamp()== null || StringUtils.isBlank(req.getStamp())) {

errors.add(new Error("21", "Stamp", "Please Enter Stamp"));

} else if (req.getStamp().length() > 50) {

errors.add(new Error("21", "Stamp", "Please Enter Stamp within 50 Characters"));

}

if (req.getCurrencyName()== null || StringUtils.isBlank(req.getCurrencyName())) {

errors.add(new Error("22", "Currency Name", "Please Enter Currency Name"));

} else if (req.getCurrencyName().length() > 25) {

errors.add(new Error("22", "Currency Name", "Please Enter Currency Name within 25 Characters"));

}

if (req.getCurrencyAbbreviation()== null || StringUtils.isBlank(req.getCurrencyAbbreviation())) {

errors.add(new Error("23", "Currency Abbreviation", "Please Enter Currency Abbreviation"));

} else if (req.getCurrencyAbbreviation().length() > 5) {

errors.add(new Error("23", "Currency Abbreviation", "Please Enter Currency Abbreviation within 5 Characters"));

}

if (req.getCurrencyAcronym()== null || StringUtils.isBlank(req.getCurrencyAcronym())) {

errors.add(new Error("24", "Currency Acronym", "Please Enter Currency Acronym"));

} else if (req.getCurrencyAcronym().length() > 5) {

errors.add(new Error("24", "Currency Acronym", "Please Enter Currency Acronym within 5 Characters"));

}

if (req.getOriginationCountryId()== null || StringUtils.isBlank(req.getOriginationCountryId())) {

errors.add(new Error("25", "Origination Country Id()", "Please Enter Origination Country Id"));

}

if (req.getDestinationCountryId()== null || StringUtils.isBlank(req.getDestinationCountryId())) {

errors.add(new Error("26", "Destination Country Id()", "Please Enter Destination Country Id"));

}

if (req.getCurrencyDecimalName()== null || StringUtils.isBlank(req.getCurrencyDecimalName())) {

errors.add(new Error("27", "Currency Decimal Name", "Please Enter Currency Decimal Name"));

} else if (req.getCurrencyDecimalName().length() > 20) {

errors.add(new Error("27", "Currency Decimal Name", "Please Enter Currency Decimal Name within 20 Characters"));

}

if (req.getCurrencyDecimalDigit()== null || StringUtils.isBlank(req.getCurrencyDecimalDigit())) {

errors.add(new Error("28", "Currency Decimal Digit", "Please Enter Currency Decimal Digit"));

}

if (req.getDecimalPlaces()==null || StringUtils.isBlank(req.getDecimalPlaces())) {

errors.add(new Error("29", "Decimal Places", "Please Enter Decimal Places"));

}

if (req.getTax()==null || StringUtils.isBlank(req.getTax())) {

errors.add(new Error("30", "Tax", "Please Enter Tax"));

}

if (req.getBranchPrefix()== null || StringUtils.isBlank(req.getBranchPrefix())) {

errors.add(new Error("31", "Branch Prefix", "Please Enter Branch Prefix"));

} else if (req.getBranchPrefix().length() > 5) {

errors.add(new Error("31", "Branch Prefix", "Please Enter Branch Prefix within 5 Characters"));

}

if (req.getPckeyCloseTrn()== null || StringUtils.isBlank(req.getPckeyCloseTrn())) {

errors.add(new Error("32", "PckeyCloseTrn", "Please Enter PckeyCloseTrn"));

} else if (req.getPckeyCloseTrn().length() > 5) {

errors.add(new Error("32", "PckeyCloseTrn", "Please Enter PckeyCloseTrn within 5 Characters"));

}

if (req.getLongitudeYn()== null || StringUtils.isBlank(req.getLongitudeYn())) {

errors.add(new Error("33", "Longitude", "Please Enter Longitude"));

} else if (req.getLongitudeYn().length() > 1) {

errors.add(new Error("33", "Longitude", "Please Enter Longitude within 1 Character"));

} else if (!(req.getLongitudeYn().equals("Y") || req.getLongitudeYn().equals("N"))) {

errors.add(new Error("33", "Longitude", "Please Enter Longitude Properly Yes-Y or No-N"));

}

if (req.getDeptCode()== null || StringUtils.isBlank(req.getDeptCode())) {

errors.add(new Error("34", "Dept Code", "Please Enter Dept Code"));

} else if (req.getDeptCode().length() > 10) {

errors.add(new Error("34", "DeptCode", "Please Enter DeptCode within 10 Characters"));

}

if (req.getExchangeId()== null || StringUtils.isBlank(req.getExchangeId())) {

errors.add(new Error("35", "ExchangeId", "Please Enter ExchangeId"));

}

}

}

catch (Exception e) {

e.printStackTrace();

log.info("Exception is --->" + e.getMessage());

return errors;

}

return errors;

}

/////// Save and Update

@Override

public CrmBranchMasterSuccessRes saveCrmBranchMaster(CrmBranchMasterSaveReq req, BranchFiles bf) {

CrmBranchMasterSuccessRes res = new CrmBranchMasterSuccessRes();

CrmBranchMaster entity = new CrmBranchMaster();

SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy");

ModelMapper mapper = new ModelMapper();

Date entryDate = null;

Double branchCode = 0D;

try {

if (StringUtils.isBlank(req.getBranchCode())) {

CrmBranchMaster crmBranchMaster = repository.findTop1ByOrderByBranchCodeDesc();

branchCode = Double.valueOf( crmBranchMaster.getBranchCode() + 1);

entryDate = new Date();

res= mapper.map(req, CrmBranchMasterSuccessRes.class);

res.setResponse("Saved Successfully ");

res.setBranchcode(decimalFormat.format(branchCode));

}

else {

// Update

branchCode = Double.valueOf(req.getBranchCode());

CrmBranchMaster data = repository.findTop1ByBranchCodeOrderByEffectiveDateDesc(branchCode);

entryDate = data.getEntryDate();

res.setResponse("Updated Successfully ");

res.setBranchcode(decimalFormat.format(branchCode));

}

// Upload Images

String res1="";

MultipartFile headerfile = bf.getHeaderfile();

MultipartFile footerfile = bf.getFooterfile();

MultipartFile signfile = bf.getSignfile();

MultipartFile stampfile = bf.getStampfile();

Random random = new Random();

String newfilename, fileextension,newfilename1;

String headerfilename, footerfilename, signfilename, stampfilename ;

String headerfilename1= "";

String footerfilename1= "";

String signfilename1 = "";

String stampfilename1 ="";

File file1;

Path destination;

// Save Original Image

destination = Paths.get(directoryPath) ;

// (i) Header Image

headerfilename= random.nextInt(100) + generateFileName()+"."+FilenameUtils.getExtension(headerfile.getOriginalFilename());

Files.copy(headerfile.getInputStream(),destination.resolve(headerfilename));

fileextension = FilenameUtils.getExtension(headerfile.getOriginalFilename());

// (ii) Footer Image

footerfilename= random.nextInt(100) + generateFileName()+"."+FilenameUtils.getExtension(footerfile.getOriginalFilename());

Files.copy(footerfile.getInputStream(),destination.resolve(footerfilename));

fileextension = FilenameUtils.getExtension(footerfile.getOriginalFilename());

// (iii) Sign Image

signfilename= random.nextInt(100) + generateFileName()+"."+FilenameUtils.getExtension(signfile.getOriginalFilename());

Files.copy(signfile.getInputStream(),destination.resolve(signfilename));

fileextension = FilenameUtils.getExtension(signfile.getOriginalFilename());

//(iv) Stamp

stampfilename= random.nextInt(100) + generateFileName()+"."+FilenameUtils.getExtension(stampfile.getOriginalFilename());

Files.copy(stampfile.getInputStream(),destination.resolve(stampfilename));

fileextension = FilenameUtils.getExtension(stampfile.getOriginalFilename());

//File Name (Ex:Jpg)

// Save Compress Header Image

if(fileextension.equals("bmp") || fileextension.equals("jpg") || fileextension.equals("jpeg")) {

headerfilename1= compressedPath + random.nextInt(100) + generateFileName()+"."+FilenameUtils.getExtension(headerfile.getOriginalFilename());

file1 = new File(directoryPath+headerfilename);

CompressImage(file1,headerfilename1);

}else {

headerfilename1 = directoryPath+headerfilename;

}

// Save Compress Header Image

if(fileextension.equals("bmp") || fileextension.equals("jpg") || fileextension.equals("jpeg")) {

footerfilename1= compressedPath + random.nextInt(100) + generateFileName()+"."+FilenameUtils.getExtension(footerfile.getOriginalFilename());

file1 = new File(directoryPath+footerfilename);

CompressImage(file1,footerfilename1);

}else {

footerfilename1 = directoryPath+footerfilename;

}

// Save Compress Sign Image

if(fileextension.equals("bmp") || fileextension.equals("jpg") || fileextension.equals("jpeg")) {

signfilename1= compressedPath + random.nextInt(100) + generateFileName()+"."+FilenameUtils.getExtension(signfile.getOriginalFilename());

file1 = new File(directoryPath+signfilename);

CompressImage(file1,signfilename1);

}else {

signfilename1 = directoryPath+signfilename;

}

// Save Compress Stamp Image

if(fileextension.equals("bmp") || fileextension.equals("jpg") || fileextension.equals("jpeg")) {

stampfilename1= compressedPath + random.nextInt(100) + generateFileName()+"."+FilenameUtils.getExtension(stampfile.getOriginalFilename());

file1 = new File(directoryPath+stampfilename);

CompressImage(file1,stampfilename1);

}else {

stampfilename1 = directoryPath+stampfilename;

}

entity.setBranchCode(branchCode);

entity.setEntryDate(entryDate);

entity.setAddress1(req.getAddress1());

entity.setAddress2(req.getAddress2());

entity.setAddress3(req.getAddress3());

entity.setBelongingBranch(req.getBelongingBranch());

entity.setBelongingType(req.getBelongingType());

entity.setBranchName(req.getBranchName());

entity.setBranchPrefix(req.getBranchPrefix());

entity.setCity(req.getCity());

entity.setCoreAppCode(req.getCoreAppCode());

entity.setCountry(req.getCountry());

entity.setCurrencyAbbreviation(req.getCurrencyAbbreviation());

entity.setCurrencyAcronym(req.getCurrencyAcronym());

entity.setCurrencyDecimalDigit(new BigDecimal(Double.valueOf(req.getCurrencyDecimalDigit())));

entity.setCurrencyDecimalName(req.getCurrencyDecimalName());

entity.setCurrencyName(req.getCurrencyName());

entity.setDecimalPlaces(new BigDecimal(Double.valueOf(req.getDecimalPlaces())));

entity.setDeptCode(req.getDeptCode());

entity.setDestinationCountryId(new BigDecimal(Double.valueOf(req.getDestinationCountryId())));

entity.setEffectiveDate(sdf.parse(req.getEffectiveDate()));

entity.setEmail(req.getEmail());

entity.setExchangeId(new BigDecimal(Integer.valueOf(req.getExchangeId())));

entity.setFax(new BigDecimal(req.getFax()));

entity.setFooterImg(footerfilename);

entity.setHeaderCompressedImg(headerfilename1);

entity.setHeaderOriginalImg(headerfilename);

entity.setFooterCompressedImg(footerfilename1);

entity.setFooterOriginalImg(footerfilename);

entity.setSignCompressedImg(signfilename1);

entity.setSignOriginalImg(signfilename);

entity.setStampCompressedImg(stampfilename1);

entity.setStampOriginalImg(stampfilename);

entity.setInsCompanyId(Double.valueOf(req.getInsCompanyId()));

entity.setLongitudeYn(req.getLongitudeYn());

entity.setOriginationCountryId(new BigDecimal(Double.valueOf(req.getOriginationCountryId())));

entity.setPckeyCloseTrn(req.getPckeyCloseTrn());

entity.setPhone(req.getPhone());

entity.setRegionCode(req.getRegionCode());

entity.setRemarks(req.getRemarks());

entity.setSignImg(signfilename);

entity.setStamp(stampfilename);

entity.setHeaderImg(headerfilename);

entity.setStatus(req.getStatus());

entity.setTax(new BigDecimal(Double.valueOf(req.getTax())));

repository.save(entity);

log.info("Saved Details is ---> " + json.toJson(entity));

} catch (Exception ex) {

log.error(ex);

return null;

}

return res;

}

private String generateFileName() {

SimpleDateFormat sdf = new SimpleDateFormat("ddMMyyyyhmmssSSSSSSa");

Calendar cal = Calendar.getInstance();

String date = sdf.format(cal.getTime());

return date;

}

public void CompressImage(File uploadFile, String documentPath) {

try {

String extension = FilenameUtils.getExtension(documentPath);

File jpgoutput = new File("thumbnail." + extension);

BufferedImage originalImage = ImageIO.read(uploadFile);

Thumbnails.of(originalImage).size(750, 750).outputFormat(extension).toFile(jpgoutput);

FileUtils.copyFile(jpgoutput, new File(documentPath));

if(jpgoutput.exists()) {

log.info("Thumbnail File Deleted after conversion");

FileUtils.deleteQuietly(jpgoutput);

}

} catch (IOException e) {

e.printStackTrace();

try {

FileUtils.copyFile(uploadFile, new File(documentPath));

} catch (IOException ex) {

ex.printStackTrace();

}

}

}

/////////// Get All

@Override

public List<CrmBranchMasterRes> getAllCrmBranchMaster() {

List<CrmBranchMasterRes> resList = new ArrayList<CrmBranchMasterRes>();

ModelMapper mapper = new ModelMapper();

try {

SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy");

List<CrmBranchMaster> crmbranchmaster = repository.OrderByBranchCodeAsc();

for (CrmBranchMaster data : crmbranchmaster) {

CrmBranchMasterRes res = new CrmBranchMasterRes();

res.setEffectiveDate(data.getEffectiveDate()==null?"":sdf.format( data.getEffectiveDate()));

res.setAddress1(data.getAddress1()== null ?"": data.getAddress1());

res.setAddress2(data.getAddress2() == null ?"": data.getAddress2());;

res.setAddress3(data.getAddress3() == null ?"": data.getAddress3());

res.setBelongingBranch(data.getBelongingBranch()==null ? "" : data.getBelongingBranch());

res.setBelongingType(data.getBelongingType() ==null ? "" : data.getBelongingType());

res.setBranchCode(data.getBranchCode()== null ?"" : decimalFormat.format(data.getBranchCode()));

res.setBranchName(data.getBranchName()== null ?"" : data.getBranchName());

res.setBranchPrefix(data.getBranchPrefix() == null ?"": data.getBranchPrefix());

res.setCity(data.getCity() == null ?"":data.getCity());

res.setCoreAppCode(data.getCoreAppCode() == null?"": data.getCoreAppCode());

res.setCountry(data.getCountry()==null?"":data.getCountry());

res.setCurrencyAbbreviation(data.getCurrencyAbbreviation()==null ?"": data.getCurrencyAbbreviation());

res.setCurrencyAcronym(data.getCurrencyAcronym()==null ?"": data.getCurrencyAcronym());

res.setCurrencyDecimalDigit(data.getCurrencyDecimalDigit()== null ?"": decimalFormat.format(data.getCurrencyDecimalDigit()));

res.setCurrencyDecimalName(data.getCurrencyDecimalName() == null ?"": data.getCurrencyDecimalName());

res.setCurrencyName(data.getCurrencyName()== null ?"": data.getCurrencyName());

res.setDecimalPlaces(data.getDecimalPlaces()== null ?"":decimalFormat.format(data.getDecimalPlaces()));

res.setDeptCode(data.getDeptCode()==null ?"": data.getDeptCode());

res.setDestinationCountryId(data.getDestinationCountryId() == null ?"":decimalFormat.format(data.getDestinationCountryId()));

res.setEmail(data.getEmail()==null ?"": data.getEmail());

res.setEntryDate(data.getEntryDate()==null ?"": sdf.format(data.getEntryDate()));

res.setExchangeId(data.getExchangeId()==null ?"":decimalFormat.format(data.getExchangeId()));

res.setFax(data.getFax()==null ?"":decimalFormat.format(data.getFax()));

res.setFooterImg(data.getFooterImg()==null ?"": data.getFooterImg());

res.setHeaderImg(data.getHeaderImg()==null ?"": data.getHeaderImg());

res.setInsCompanyId(data.getInsCompanyId()==null?"":decimalFormat.format(data.getInsCompanyId()));

res.setLongitudeYn(data.getLongitudeYn()==null ?"": data.getLongitudeYn());

res.setOriginationCountryId(data.getOriginationCountryId()==null?"" :decimalFormat.format(data.getOriginationCountryId()));

res.setPckeyCloseTrn(data.getPckeyCloseTrn()==null ?"": data.getPckeyCloseTrn());

res.setPhone(data.getPhone()==null ?"": data.getPhone());

res.setRegionCode(data.getRegionCode() == null ?"": data.getRegionCode());

res.setRemarks(data.getRemarks()==null ?"": data.getRemarks());

res.setSignImg(data.getSignImg()==null ?"": data.getSignImg());

res.setStamp(data.getStamp()== null ?"": data.getStamp());

res.setStatus(data.getStatus()== null ?"": data.getStatus());

res.setTax(data.getTax()== null ?"": decimalFormat.format(data.getTax()));

resList.add(res);

}

} catch (Exception e) {

e.printStackTrace();

log.info("Exception is ---> " + e.getMessage());

return null;

}

return resList;

}

//////// Get By branch code

@Override

public CrmBranchMasterRes getCrmBranchMaster(String branchCode) {

CrmBranchMasterRes res = new CrmBranchMasterRes();

ModelMapper mapper = new ModelMapper();

try {

SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy");

CrmBranchMaster data = repository.findByBranchCode(Double.valueOf(branchCode));

res.setEffectiveDate(data.getEffectiveDate()==null?"":sdf.format( data.getEffectiveDate()));

res.setAddress1(data.getAddress1()== null ?"": data.getAddress1());

res.setAddress2(data.getAddress2() == null ?"": data.getAddress2());

res.setAddress3(data.getAddress3() == null ?"": data.getAddress3());

res.setBelongingBranch(data.getBelongingBranch()==null ? "" : data.getBelongingBranch());

res.setBelongingType(data.getBelongingType() ==null ? "" : data.getBelongingType());

res.setBranchCode(data.getBranchCode()== null ?"" : decimalFormat.format(data.getBranchCode()));

res.setBranchName(data.getBranchName()== null ?"" : data.getBranchName());

res.setBranchPrefix(data.getBranchPrefix() == null ?"": data.getBranchPrefix());

res.setCity(data.getCity() == null ?"":data.getCity());

res.setCoreAppCode(data.getCoreAppCode() == null?"": data.getCoreAppCode());

res.setCountry(data.getCountry()==null?"":data.getCountry());

res.setCurrencyAbbreviation(data.getCurrencyAbbreviation()==null ?"": data.getCurrencyAbbreviation());

res.setCurrencyAcronym(data.getCurrencyAcronym()==null ?"": data.getCurrencyAcronym());

res.setCurrencyDecimalDigit(data.getCurrencyDecimalDigit()== null ?"": decimalFormat.format(data.getCurrencyDecimalDigit()));

res.setCurrencyDecimalName(data.getCurrencyDecimalName() == null ?"": data.getCurrencyDecimalName());

res.setCurrencyName(data.getCurrencyName()== null ?"": data.getCurrencyName());

res.setDecimalPlaces(data.getDecimalPlaces()== null ?"":decimalFormat.format(data.getDecimalPlaces()));

res.setDeptCode(data.getDeptCode()==null ?"": data.getDeptCode());

res.setDestinationCountryId(data.getDestinationCountryId() == null ?"":decimalFormat.format(data.getDestinationCountryId()));

res.setEmail(data.getEmail()==null ?"": data.getEmail());

res.setEntryDate(data.getEntryDate()==null ?"": sdf.format(data.getEntryDate()));

res.setExchangeId(data.getExchangeId()==null ?"":decimalFormat.format(data.getExchangeId()));

res.setFax(data.getFax()==null ?"":decimalFormat.format(data.getFax()));

res.setInsCompanyId(data.getInsCompanyId()==null?"":decimalFormat.format(data.getInsCompanyId()));

res.setLongitudeYn(data.getLongitudeYn()==null ?"": data.getLongitudeYn());

res.setOriginationCountryId(data.getOriginationCountryId()==null?"" :decimalFormat.format(data.getOriginationCountryId()));

res.setPckeyCloseTrn(data.getPckeyCloseTrn()==null ?"": data.getPckeyCloseTrn());

res.setPhone(data.getPhone()==null ?"": data.getPhone());

res.setRegionCode(data.getRegionCode() == null ?"": data.getRegionCode());

res.setRemarks(data.getRemarks()==null ?"": data.getRemarks());

res.setStatus(data.getStatus()== null ?"": data.getStatus());

res.setTax(data.getTax()== null ?"": decimalFormat.format(data.getTax()));

res.setHeaderCompressedImg(data.getHeaderCompressedImg());

res.setFooterCompressedImg(data.getFooterCompressedImg());

res.setSignCompressedImg(data.getSignCompressedImg());

res.setStampCompressedImg(data.getStampCompressedImg());

res.setHeaderOriginalImg(data.getHeaderOriginalImg());

res.setFooterOriginalImg(data.getFooterOriginalImg());

res.setSignOriginalImg(data.getSignOriginalImg());

res.setStampOriginalImg(data.getStampOriginalImg());

if (StringUtils.isNotBlank(data.getHeaderCompressedImg()) && new File(data.getHeaderCompressedImg()).exists())

{

res.setHeaderImg(new GetFileFromPath(data.getHeaderCompressedImg()).call().getImgUrl());

}

if (StringUtils.isNotBlank(data.getFooterCompressedImg()) && new File(data.getFooterCompressedImg()).exists())

{

res.setFooterImg(new GetFileFromPath(data.getFooterCompressedImg()).call().getImgUrl());

}

if (StringUtils.isNotBlank(data.getSignCompressedImg()) && new File(data.getSignCompressedImg()).exists())

{

res.setSignImg(new GetFileFromPath(data.getSignCompressedImg()).call().getImgUrl());

}

if (StringUtils.isNotBlank(data.getStampCompressedImg()) && new File(data.getStampCompressedImg()).exists())

{

res.setStamp(new GetFileFromPath(data.getStampCompressedImg()).call().getImgUrl());

}

}

catch (Exception e) {

e.printStackTrace();

log.info("Exception is ---> " + e.getMessage());

return null;

}

return res;

}

/// Get By Active status

@Override

public List<CrmBranchMasterRes> getCrmBranchMasterActive() {

List<CrmBranchMasterRes> resList = new ArrayList<CrmBranchMasterRes>();

ModelMapper mapper = new ModelMapper();

try {

SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy");

List<CrmBranchMaster> crmbranchmaster = repository.findByStatusOrderByBranchCodeAsc("Y");

for (CrmBranchMaster data : crmbranchmaster) {

CrmBranchMasterRes res = new CrmBranchMasterRes();

res.setEffectiveDate(data.getEffectiveDate()==null?"":sdf.format( data.getEffectiveDate()));

res.setAddress1(data.getAddress1()== null ?"": data.getAddress1());

res.setAddress2(data.getAddress2() == null ?"": data.getAddress2());

res.setAddress3(data.getAddress3() == null ?"": data.getAddress3());

res.setBelongingBranch(data.getBelongingBranch()==null ? "" : data.getBelongingBranch());

res.setBelongingType(data.getBelongingType() ==null ? "" : data.getBelongingType());

res.setBranchCode(data.getBranchCode()== null ?"" : decimalFormat.format(data.getBranchCode()));

res.setBranchName(data.getBranchName()== null ?"" : data.getBranchName());

res.setBranchPrefix(data.getBranchPrefix() == null ?"": data.getBranchPrefix());

res.setCity(data.getCity() == null ?"":data.getCity());

res.setCoreAppCode(data.getCoreAppCode() == null?"": data.getCoreAppCode());

res.setCountry(data.getCountry()==null?"":data.getCountry());

res.setCurrencyAbbreviation(data.getCurrencyAbbreviation()==null ?"": data.getCurrencyAbbreviation());

res.setCurrencyAcronym(data.getCurrencyAcronym()==null ?"": data.getCurrencyAcronym());

res.setCurrencyDecimalDigit(data.getCurrencyDecimalDigit()== null ?"": decimalFormat.format(data.getCurrencyDecimalDigit()));

res.setCurrencyDecimalName(data.getCurrencyDecimalName() == null ?"": data.getCurrencyDecimalName());

res.setCurrencyName(data.getCurrencyName()== null ?"": data.getCurrencyName());

res.setDecimalPlaces(data.getDecimalPlaces()== null ?"":decimalFormat.format(data.getDecimalPlaces()));

res.setDeptCode(data.getDeptCode()==null ?"": data.getDeptCode());

res.setDestinationCountryId(data.getDestinationCountryId() == null ?"":decimalFormat.format(data.getDestinationCountryId()));

res.setEmail(data.getEmail()==null ?"": data.getEmail());

res.setEntryDate(data.getEntryDate()==null ?"": sdf.format(data.getEntryDate()));

res.setExchangeId(data.getExchangeId()==null ?"":decimalFormat.format(data.getExchangeId()));

res.setFax(data.getFax()==null ?"":decimalFormat.format(data.getFax()));

res.setFooterImg(data.getFooterImg()==null ?"": data.getFooterImg());

res.setHeaderImg(data.getHeaderImg()==null ?"": data.getHeaderImg());

res.setInsCompanyId(data.getInsCompanyId()==null?"":decimalFormat.format(data.getInsCompanyId()));

res.setLongitudeYn(data.getLongitudeYn()==null ?"": data.getLongitudeYn());

res.setOriginationCountryId(data.getOriginationCountryId()==null?"" :decimalFormat.format(data.getOriginationCountryId()));

res.setPckeyCloseTrn(data.getPckeyCloseTrn()==null ?"": data.getPckeyCloseTrn());

res.setPhone(data.getPhone()==null ?"": data.getPhone());

res.setRegionCode(data.getRegionCode() == null ?"": data.getRegionCode());

res.setRemarks(data.getRemarks()==null ?"": data.getRemarks());

res.setSignImg(data.getSignImg()==null ?"": data.getSignImg());

res.setStamp(data.getStamp()== null ?"": data.getStamp());

res.setStatus(data.getStatus()== null ?"": data.getStatus());

res.setTax(data.getTax()== null ?"": decimalFormat.format(data.getTax()));

resList.add(res);

}

} catch (Exception e) {

e.printStackTrace();

log.info("Exception is ---> " + e.getMessage());

return null;

}

return resList;

}

/////// Pagination

@Override

public List<CrmBranchMasterRes> getAllPaginationCrmBranchMaster(CrmBranchMasterPaginationReq req) {

List<CrmBranchMasterRes> resList = new ArrayList<CrmBranchMasterRes>();

ModelMapper mapper = new ModelMapper();

try {

SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy");

int limit =StringUtils.isBlank(req.getLimit())?0:Integer.valueOf(req.getLimit());

int offset = StringUtils.isBlank(req.getOffset())?10:Integer.valueOf(req.getOffset());

Pageable paging = PageRequest.of(limit, offset, Sort.by("entryDate").descending());

Page<CrmBranchMaster> crmbranchmaster = repository.findAll(paging);

// Page<CrmBranchMaster> crmbranchmaster = repository.findByStatus(paging , "Y");

for (CrmBranchMaster data : crmbranchmaster) {

CrmBranchMasterRes res = new CrmBranchMasterRes();

res.setEffectiveDate(data.getEffectiveDate()==null?"":sdf.format( data.getEffectiveDate()));

res.setAddress1(data.getAddress1()== null ?"": data.getAddress1());

res.setAddress2(data.getAddress2() == null ?"": data.getAddress2());;

res.setAddress3(data.getAddress3() == null ?"": data.getAddress3());

res.setBelongingBranch(data.getBelongingBranch()==null ? "" : data.getBelongingBranch());

res.setBelongingType(data.getBelongingType() ==null ? "" : data.getBelongingType());

res.setBranchCode(data.getBranchCode()== null ?"" : decimalFormat.format(data.getBranchCode()));

res.setBranchName(data.getBranchName()== null ?"" : data.getBranchName());

res.setBranchPrefix(data.getBranchPrefix() == null ?"": data.getBranchPrefix());

res.setCity(data.getCity() == null ?"":data.getCity());

res.setCoreAppCode(data.getCoreAppCode() == null?"": data.getCoreAppCode());

res.setCountry(data.getCountry()==null?"":data.getCountry());

res.setCurrencyAbbreviation(data.getCurrencyAbbreviation()==null ?"": data.getCurrencyAbbreviation());

res.setCurrencyAcronym(data.getCurrencyAcronym()==null ?"": data.getCurrencyAcronym());

res.setCurrencyDecimalDigit(data.getCurrencyDecimalDigit()== null ?"": decimalFormat.format(data.getCurrencyDecimalDigit()));

res.setCurrencyDecimalName(data.getCurrencyDecimalName() == null ?"": data.getCurrencyDecimalName());

res.setCurrencyName(data.getCurrencyName()== null ?"": data.getCurrencyName());

res.setDecimalPlaces(data.getDecimalPlaces()== null ?"":decimalFormat.format(data.getDecimalPlaces()));

res.setDeptCode(data.getDeptCode()==null ?"": data.getDeptCode());

res.setDestinationCountryId(data.getDestinationCountryId() == null ?"":decimalFormat.format(data.getDestinationCountryId()));

res.setEmail(data.getEmail()==null ?"": data.getEmail());

res.setEntryDate(data.getEntryDate()==null ?"": sdf.format(data.getEntryDate()));

res.setExchangeId(data.getExchangeId()==null ?"":decimalFormat.format(data.getExchangeId()));

res.setFax(data.getFax()==null ?"":decimalFormat.format(data.getFax()));

res.setFooterImg(data.getFooterImg()==null ?"": data.getFooterImg());

res.setHeaderImg(data.getHeaderImg()==null ?"": data.getHeaderImg());

res.setInsCompanyId(data.getInsCompanyId()==null?"":decimalFormat.format(data.getInsCompanyId()));

res.setLongitudeYn(data.getLongitudeYn()==null ?"": data.getLongitudeYn());

res.setOriginationCountryId(data.getOriginationCountryId()==null?"" :decimalFormat.format(data.getOriginationCountryId()));

res.setPckeyCloseTrn(data.getPckeyCloseTrn()==null ?"": data.getPckeyCloseTrn());

res.setPhone(data.getPhone()==null ?"": data.getPhone());

res.setRegionCode(data.getRegionCode() == null ?"": data.getRegionCode());

res.setRemarks(data.getRemarks()==null ?"": data.getRemarks());

res.setSignImg(data.getSignImg()==null ?"": data.getSignImg());

res.setStamp(data.getStamp()== null ?"": data.getStamp());

res.setStatus(data.getStatus()== null ?"": data.getStatus());

res.setTax(data.getTax()== null ?"": decimalFormat.format(data.getTax()));

resList.add(res);

}

} catch (Exception e) {

e.printStackTrace();

log.info("Exception is ---> " + e.getMessage());

return null;

}

return resList;

}

}

/\*

\* public CrmBranchMasterServiceImpl(CrmBranchMasterRepository repo) {

\* this.repository = repo; }

\*

\*/

/\*

\* @Override public CrmBranchMaster create(CrmBranchMaster d) {

\*

\* CrmBranchMaster entity;

\*

\* try { entity = repository.save(d);

\*

\* } catch (Exception ex) { log.error(ex); return null; } return entity; }

\*

\*

\* @Override public CrmBranchMaster update(CrmBranchMaster d) { CrmBranchMaster

\* c;

\*

\* try { c = repository.saveAndFlush(d);

\*

\* } catch (Exception ex) { log.error(ex); return null; } return c; }

\*

\*

\* @Override public CrmBranchMaster getOne(long id) { CrmBranchMaster t;

\*

\* try { t = repository.findById(id).orElse(null);

\*

\* } catch (Exception ex) { log.error(ex); return null; } return t; }

\*

\*

\* @Override public List<CrmBranchMaster> getAll() { List<CrmBranchMaster> lst;

\*

\* try { lst = repository.findAll();

\*

\* } catch (Exception ex) { log.error(ex); return Collections.emptyList(); }

\* return lst; }

\*

\*

\* @Override public long getTotal() { long total;

\*

\* try { total = repository.count(); } catch (Exception ex) { log.error(ex);

\* return 0; } return total; }

\*

\*

\* @Override public CommonCrmRes getCrmBranches(BranchDropDownReq req) {

\* CommonCrmRes commonRes = new CommonCrmRes(); List<DropDownRes> resList = new

\* ArrayList<DropDownRes>(); try { LoginMaster loginData =

\* loginRepo.findByLoginIdAndCompanyId(req.getLoginId(),

\* Double.valueOf(req.getInsuranceId()));

\*

\* if (loginData.getAttachedBranch() != null) { // Find Branches List<String>

\* rsaBranchCodes = Arrays.asList(loginData.getAttachedBranch().split(","));

\* List<CrmBranchMaster> branches =

\* repository.findByStatusAndBranchCodeInOrderByBranchNameAsc("Y",

\* rsaBranchCodes);

\*

\* // Response for (CrmBranchMaster data : branches) { DropDownRes res = new

\* DropDownRes(); res.setCode(data.getBranchCode());

\* res.setCodedesc(data.getBranchName()); resList.add(res); } }

\*

\* // Response commonRes.setCommonResponse(resList);

\* commonRes.setIsError(false); commonRes.setMessage("Success");

\*

\* } catch (Exception e) { e.printStackTrace(); log.info("Exception is ---> " +

\* e.getMessage()); commonRes.setCommonResponse(null);

\* commonRes.setIsError(true); commonRes.setMessage("Failed");

\*

\* return commonRes; } return commonRes; }

\*

\*

\* @Override public CommonCrmRes getCrmBranchesWithOther(BranchDropDownReq req)

\* { CommonCrmRes commonRes = new CommonCrmRes(); List<DropDownRes> resList =

\* new ArrayList<DropDownRes>(); try { LoginMaster loginData =

\* loginRepo.findByLoginIdAndCompanyId(req.getLoginId(),

\* Double.valueOf(req.getInsuranceId()));

\*

\* if (loginData.getAttachedBranch() != null) { // Find Branches List<String>

\* rsaBranchCodes = Arrays.asList(loginData.getAttachedBranch().split(","));

\* List<CrmBranchMaster> branches =

\* repository.findByStatusAndBranchCodeInOrderByBranchNameAsc("Y",

\* rsaBranchCodes);

\*

\* // Response for (CrmBranchMaster data : branches) { DropDownRes res = new

\* DropDownRes(); res.setCode(data.getBranchCode());

\* res.setCodedesc(data.getBranchName()); resList.add(res); } }

\*

\* //OTher Key DropDownRes res = new DropDownRes(); res.setCode("Others");

\* res.setCodedesc("Others"); resList.add(res);

\*

\* // Response commonRes.setCommonResponse(resList);

\* commonRes.setIsError(false); commonRes.setMessage("Success");

\*

\* } catch (Exception e) { e.printStackTrace(); log.info("Exception is ---> " +

\* e.getMessage()); commonRes.setCommonResponse(null);

\* commonRes.setIsError(true); commonRes.setMessage("Failed");

\*

\* return commonRes; } return commonRes; }

\*

\*

\* @Override public boolean delete(long id) { try { repository.deleteById(id);

\* return true;

\*

\* } catch (Exception ex) { log.error(ex); return false; } }

\*

\*

\*

\*/